


# NETWORK COMPUTER SYSTEM AND SUBSTITUTE COMPILING SERVER DEVICE

Patent number: JP2000215181  
 Publication date: 2000-08-04  
 Inventor: SHIMURA HIROYA  
 Applicant: FUJITSU LTD  
 Classification:  
 - international: G06F15/16; G06F9/45; G06F9/46  
 - european:  
 Application number: JP19990013011 19990121  
 Priority number(s):

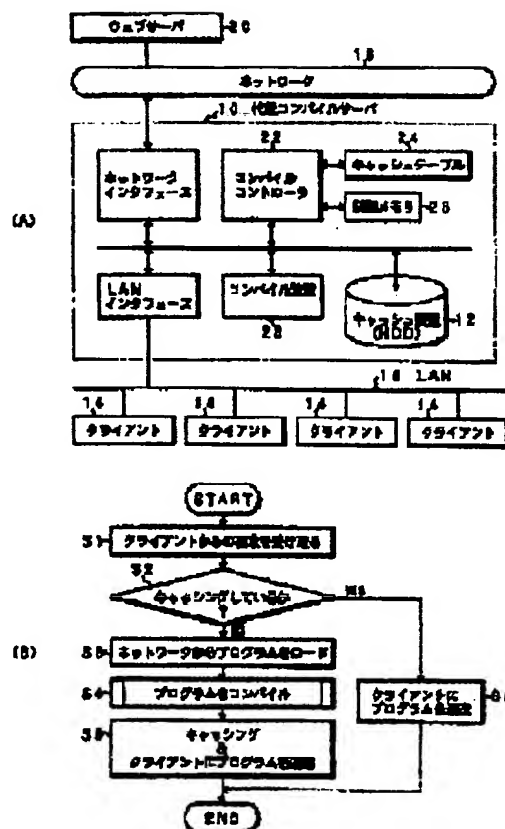
Also published as:  
 US6370687 (B1)

Report a data error here

## Abstract of JP2000215181

**PROBLEM TO BE SOLVED:** To enable a client to fast execute a virtual machine computer program in a network by giving a compiling function to a substitute server.

**SOLUTION:** A client 14 is prepared to execute a virtual machine computer program in a network 18 together with a substitute compiling server 10 which receives the virtual machine computer program from the network 18 and transfers it to the client 14, i.e., a requester after compiling the program. Thus, a compiling function of the virtual machine computer program is given to a substitute server in the network 18 to obtain the server 10. As a result, the burden of the client 14 which executes the virtual machine computer program is reduced and the client 14 can fast execute an application program in a network computer system.



BEST AVAILABLE COPY



とのできる等の理由により急速に広まっている。

【0003】このようなJava言語で記述されたJavaプログラムは、プラットフォームを選ばないJava仮想マシン(JVM: Java Virtual Machine)のプログラムであるバイトコードとして配布されるため、クライアントの計算機では直接実行することができず、Javaプログラムの実行は、クライアント側でJava仮想マシンのコードをエミュレートする必要がある、インタプリタ方式で実行される。

【0004】インタプリタ方式は、ソフトウェアによりバイトコードを逐次解釈して実行するエミュレーションであり、他の言語と比較して実行が遅くなる。これを解決する方法としてJITコンパイラ(Just In Time Compiler)をクライアント側で使う手法がある。JITコンパイラは、バイトコードのプログラムをネットワークからロードして実行する前に、バイトコードをクライアントマシンのネイティブコードにコンパイルすることにより、直接、クライアントのハードウェアで実行でき、高速な実行が可能となる。

【0005】図1に従来のウェブサーバ(Web Server)のアクセスにプロキシサーバ(Proxy Server)を使ったシステムの構成を示す。プロキシサーバを使った場合の基本的な処理の流れは次のようになる。LAN102に接続されたクライアント100-1~100-4は、通常、httpプロトコルを用いてプロキシサーバ104にウェブサーバ110のページ、イメージ、プログラム等を取ってくるように要求を出す。

【0006】例えばクライアント100-2が要求を出すと、プロキシサーバ104は要求されたデータがハードディスク等のキャッシュ106上に保存してあるかどうか調べ、保存されていない場合、外部のネットワーク108を介してウェブサーバ110にアクセスし、必要なデータを取ってくる。

【0007】プロキシサーバ104は、ハードディスク等のキャッシュ106にウェブサーバ110から獲得したデータを保存すると同時に、要求元のクライアント100-2にデータを返す。もし、データがキャッシュ106上に保存されていた場合は、キャッシュ106からデータを取り出し、クライアント100-2に返送する。

【0008】このため、従来、ネットワーク上のウェブサーバ110に存在するJavaプログラムを例えばクライアント100-2が実行するときは、プロキシサーバ106からJavaプログラムを受け取ったクライアント100-2が、JavaプログラムをJITコンパイラでコンパイルしながら実行している。

【0009】

【発明が解決しようとする課題】しかしながら、従来のクライアント側JITコンパイラを設けた場合は、クライアントでJavaプログラムのバイトコードをコン

パイルするための時間が必要となり、プログラムの実行に影響を与える。特に、サーバからクライアントという一方のデータの流れではなく、クライアントからの要求をサーバに戻して処理したり、クライアントで処理した結果をサーバに渡す等の双方向のデータの流れをもつインタラクティブな環境では、プログラムの立ち上がり時間やレスポンスを速くすることが重要であり、コンパイルに時間をかけることはできない。

【0010】更に、プログラムの実行を高速化するためには、JITコンパイラでコンパイルを行うときに複雑な最適化を行う必要があるが、複雑な最適化を行うほどコンパイルの時間が遅くなる。また最適化を行わないと、コンパイル時間は速くなるが、実行時間は長くなるというトレードオフがある。

【0011】本発明は、代理サーバにコンパイル機能を持たせることで、クライアントでネットワーク上の仮想マシンコンピュータプログラムの高速実行を可能とするコンピュータシステム及び代理コンパイルサーバ装置を提供することを目的とする。

【0012】

【課題を解決するための手段】本発明は、図1(A)のように、ネットワーク18上の仮想マシンコンピュータプログラムを実行するクライアント14と、クライアント14の要求に基づき、ネットワーク18上のウェブサーバ20から仮想マシンコンピュータプログラム、例えばJavaプログラムを受け取り、コンパイルして要求元のクライアントに引き渡す代理コンパイルサーバ10とを備えたことを特徴とするネットワークコンピュータシステムである。

【0013】このように本発明は、高速な代理コンパイルサーバを用意し、代理コンパイルサーバにクライアント側でのJITコンパイルを肩代わりさせることにより、クライアントでのプログラムの実行を高速化させる。この場合、代理コンパイルサーバが十分高速であれば、コンパイル時に複雑な最適化を行うことができ、クライアントの実行速度が更に向上する。

【0014】代理コンパイルサーバ10は、クライアント14とプロセッサ等の相違で実行形式が異なる場合、クライアント14の実行形式に従って仮想マシンコンピュータプログラムをコンパイルする。例えば代理コンパイルサーバ10は、クライアント14の実行形式を予め把握しており、要求元のクライアント14の実行形式に従って仮想マシンコンピュータプログラムをコンパイルする。また代理コンパイルサーバ10は、クライアント14が要求した実行形式に従って仮想マシンコンピュータプログラムをコンパイルしてもよい。

【0015】更に、代理コンパイルサーバ10は、ネットワーク上のウェブサーバ20から仮想マシンコンピュータプログラムを受け取った際に、接続可能なクライアント14の実行形式の種別毎にコンパイルし、複数の実

行形式のコンパイル済みプログラムをクライアント14に受け渡し、要求元のクライアント14に自己の実行形式に合うコンパイル済みプログラムを選択させて実行させてもよい。

【0016】代理コンパイルサーバ10は、図1(B)のように、コンパイル済みのプログラムを要求元のクライアント14に引き渡すと同時にキャッシュ12上に保存しておき、クライアント14から要求があった時に、キャッシュ12上に保存しているコンパイル済みプログラムを引き渡す。

【0017】このため、代理コンパイルサーバ10が高速でなかったとしても、コンパイルしたプログラムをキャッシュ12上に保存しておくことにより、最初の実行はコンパイルに時間がかかり実行が遅くなるが、2度目からの実行は高速に実行可能となる。

【0018】代理コンパイルサーバ10は、クライアント14から要求があった際に、次にクライアント14が要求する仮想マシンコンピュータプログラムを予測してネットワーク上のウェブサーバ20から受け取り、コンパイルした後にキャッシュ12上に保存する。このような予測コンパイルによって、クライアント14の実行を更に高速化できる。

【0019】代理コンパイルサーバ10は、ネットワーク18から受け取った仮想マシンコンピュータプログラムをキャッシュ12上に保存しておき、クライアント14から要求があった時に、キャッシュ12上に保存している仮想マシンコンピュータプログラムをコンパイルしてクライアント14に受け渡す。これによってキャッシュ12の記憶容量を低減する。

【0020】この場合にも、代理コンパイルサーバ10は、クライアント14から要求があった際に、要求された仮想マシンコンピュータプログラムに加えて次にクライアント14が要求する仮想マシンコンピュータプログラムを予測してネットワーク18上のウェブサーバ20から受け取り、キャッシュ12上に保存する。

【0021】代理コンパイルサーバ10は、ネットワーク上のウェブサーバ20から仮想マシンコンピュータプログラムとして受け取ったバイトコードを、要求元のクライアント14の実行形式に適合したマシンコード(ネイティブコード)にコンパイルして引き渡す。

【0022】また本発明は、代理コンパイルサーバ10を提供することを特徴とし、代理コンパイルサーバ10は、ネットワーク18上の仮想マシンコンピュータプログラムを実行するクライアント14の要求に基づき、ネットワーク18上から要求された仮想マシンコンピュータプログラムを受け取るコントローラ22と、コントローラ22で受け取ったプログラムをコンパイルして要求元のクライアントに受け渡すコンパイル装置28とを備える。

【0023】代理コンパイルサーバ装置10は、コンパ

イル装置28でコンパイルしたプログラムを保存しておき、クライアント14から要求があった時に、コンパイル済みプログラムを受け渡すキャッシュ12を設ける。

【0024】

【発明の実施の形態】図2は、本発明のネットワークコンピュータシステムのシステム構成図である。図2において、LAN16に接続されたクライアント14-1、14-2、14-3、14-4に対しては、インターネット等のネットワーク18のウェブサーバ20とのアクセスのため代理コンパイルサーバ10が設けられている。

【0025】代理コンパイルサーバ10としては、図16と同様、プロキシサーバが使用され、このプロキシサーバにクライアント14-1~14-4がウェブサーバ20に要求した仮想マシンコンピュータプログラムであるJavaプログラムのコンパイル機能を付加することで、代理コンパイルサーバ10を実現している。代理コンパイルサーバ10にはハードディスク等を用いたキャッシュ12が接続されている。

【0026】図3は、図2のネットワークコンピュータシステムに設けた本発明による代理コンパイルサーバ10の機能ブロック図であり、ウェブサーバ及びクライアント側と共に表わしている。

【0027】プロキシサーバを用いた代理コンパイルサーバ10には、コンパイルコントローラ22、キャッシュテーブル24、制御メモリ26、コンパイル装置28、LANインタフェース30及びネットワークインタフェース32が設けられる。

【0028】ここでネットワーク18上のウェブサーバ20には、仮想マシンコンピュータプログラムとしてのJavaプログラムがウェブページ上のアプレットとして準備されており、クライアント14-1~14-4側にはホットJavaブラウザが準備され、ウェブサーバ20に対するアクセスで受け出された仮想マシンコンピュータプログラムであるJavaプログラムのバイトコードをエミュレートしてインタプリタ方式で実行する環境が設けられている。

【0029】代理コンパイルサーバ10のコンパイルコントローラ22は、クライアント14-1~14-4からhttpプロトコルを用いたJavaプログラムを取得するための要求を受けると、キャッシュテーブル24を参照し、キャッシュ装置12に要求されたJavaプログラムのコンパイルプログラムが保存されているかどうかチェックし、保存されていなければネットワーク18を経由してウェブサーバ20にJavaプログラムの引き渡しを要求する。

【0030】ウェブサーバ20から代理コンパイルサーバ10にJavaプログラムが受け渡されると、コンパイルコントローラ22はコンパイル装置28を制御し、ウェブサーバ20から受け渡されたJavaプログラム

のバイトコードを、そのとき要求を行っているクライアントの実行環境に適合したネイティブ環境のマシンコードにコンパイルし、コンパイル済みのJavaプログラムを要求元のクライアントに引き渡す。同時にキャッシュ装置12にコンパイル済みのJavaプログラムを保存し、キャッシュテーブル24にキャッシュ結果を登録する。

【0031】図4は、図3の代理コンパイルサーバ10に設けたキャッシュ装置12の一例であり、この例ではウェブサーバ20から引き渡されたコンパイル前のJavaプログラム34-1、34-2と、それぞれをコンパイル装置28でネイティブコードにコンパイルしたコンパイル済みJavaプログラム36-1、36-2を保存している。

【0032】また図3の代理コンパイルサーバ10のコンパイルコントローラ22に対して設けられた制御メモリ26には、例えば図5のように、代理コンパイルサーバ10にLAN16を介してアクセス可能なクライアント14-1~14-4のサーバ名と、ネットワークのJavaプログラムを実行するプロセッサ等で決まる実行形式の対応関係を登録している。

【0033】制御メモリ26のクライアント名A、B、C、Dは、例えばネットワークアドレス「http://WWW.A.co.jp/」等が使用され、実行形式 $\alpha$ 、 $\beta$ 、 $\gamma$ としては、クライアント14-1~14-4のプロセッサ名、例えばSPARCV8、X86、Intel486等の識別子が設定される。

【0034】この制御メモリ26により代理コンパイルサーバ10のコンパイルコントローラ22は、クライアント側からJavaプログラムの引き渡し要求を受けた際に、クライアントのサーバ名による制御メモリ26の参照でクライアントの実行形式を認識し、コンパイル装置28でJavaプログラムのバイトコードからコンパイルする実行環境のネイティブコードを認識する。

【0035】このようなクライアント側の実行形式に対応して、コンパイル装置28には図6に示すように、クライアント側の実行形式 $\alpha$ 、 $\beta$ 、 $\gamma$ に対応したJITコンパイラ(JITコンパイルプログラム)38-1、38-2、38-3が予め準備されており、要求のあったサーバ名に対応した実行形式のJITコンパイラを選択してJavaプログラムのバイトコードから対応する実行形式のネイティブコードへのコンパイルを行う。

【0036】例えばJITコンパイラ38-1はJavaバイトコードをSPARCV8のネイティブコードに翻訳するコンパイラであり、JITコンパイラ38-2はJavaバイトコードをX86のネイティブコードに翻訳するコンパイラであり、更にJITコンパイラ38-3はJavaバイトコードをIntel486のネイティブコードに翻訳するコンパイラである。

【0037】勿論、JITコンパイラ38-1~38-

3に加え、Java仮想マシンコンピュータプログラムの実行環境を提供するホットJavaブラウザがインストールされているクライアントマシンの種別に応じ、必要に応じて適宜にJITコンパイラを増設することができる。

【0038】図7は、図3の代理コンパイルサーバ10の処理動作のフローチャートである。いまステップS1で例えばクライアント14-1からウェブサーバ20に対するJavaプログラムのアクセス要求を受け取ると、ステップS2で、要求されたJavaプログラムがキャッシュ装置12に保存されているか否かチェックする。

【0039】具体的には、クライアント14-1からのJavaプログラムのアクセス要求に対しコンパイルコントローラ22が、要求を行ったクライアント名により制御メモリ26を参照して、その実行形式を認識し、続いてキャッシュテーブル24を参照し、クライアントの実行形式に適合したコンパイル済みのJavaプログラムがキャッシュ装置12上に存在するか否かチェックする。

【0040】キャッシュ装置12上に存在しなければ、コンパイルコントローラ22はネットワークインタフェース32を介してネットワーク18上のウェブサーバ20に対し、クライアント14-1から受けたプログラム要求を通知する。このプログラム要求を受けてウェブサーバ20は、該当するJavaプログラム即ちアプレットを代理コンパイルサーバ10に返送する。

【0041】ウェブサーバ20から要求したJavaプログラムを受け取った代理コンパイルサーバ10のコンパイルコントローラ22は、コンパイル装置28にJavaプログラムを引き渡すと共に、制御メモリ26の参照で認識した要求元のクライアント14-1のクライアント名に対応する実行形式に対応した図6のような複数のJITコンパイラのいずれかを指定して、ステップS4でJavaプログラムのコンパイルを行う。

【0042】コンパイル装置28により要求元クライアント14-1の実行形式のネイティブコードにメソッド単位に翻訳され且つ最適化されたコンパイル済みJavaプログラムは、キャッシュ装置12にキャッシングすると同時に、LANインタフェース30を介して要求元のクライアント14-1に返送され、クライアント14-1においてコンパイル済みのネイティブコードからなるJavaプログラムが実行される。

【0043】キャッシュ装置12に対するJavaプログラムのキャッシングは、図4に示したように、ウェブサーバ20から受け取ったコンパイル前のJavaプログラムとコンパイル装置28でコンパイルしたコンパイル済みのJavaプログラムの両方を保存する。

【0044】このうちコンパイル済みのJavaプログラムは要求元のクライアントの実行形式に対応したネイ

タイプコードのプログラムであり、キャッシングした後は、同じ実行形式のクライアントからの要求に対し応答する。

【0045】これに対し実行形式の異なるクライアントから同じJavaプログラムの要求があった場合には、キャッシュ装置12に保存しているコンパイル前のJavaプログラムを読み出し、そのときの要求元のクライアントの実行形式に対応したJITコンパイラを選択してコンパイルし、コンパイルしたJavaプログラムは要求元に返送し、同時にキャッシュ装置12に別の実行形式であるネイティブコードに翻訳されたJavaプログラムを保存する。

【0046】一方、図7のステップS2でクライアント14-1から要求されたJavaプログラムがキャッシュ装置12にキャッシングされている場合には、ステップS6に進み、キャッシュ装置12に保存していたコンパイル済みのJavaプログラムを読み出してクライアント14-1に返送する。

【0047】このようにクライアント側からの要求したJavaプログラムがキャッシュ装置14上に保存されていてキャッシュヒットとなった際には、ウェブサーバ20に対するJavaプログラムの要求及び引き渡されたJavaプログラムのコンパイルが不要となり、代理コンパイルサーバ10よりコンパイル済みのJavaプログラムを直ちに返送して、要求元のクライアントで実行することができる。

【0048】このような本発明の代理コンパイルサーバ10により、JITコンパイラの機能がサーバ側で実現され、クライアントからネットワーク上のウェブサーバに対するJavaプログラムの要求に対し、代理コンパイルサーバ10より要求元のクライアントの実行形式に適合したネイティブコードにコンパイルされ且つ最適化されたJavaプログラムが返送され、代理コンパイルサーバ10としてプロキシサーバのような高性能のマシンを使用することで、Javaプログラムのコンパイル及び最適化の処理時間がごく短時間で済み、結果としてクライアント14-1がJavaプログラムを要求してから実行するまでの処理時間を大幅に短縮し、Javaプログラムをインタプリタ方式で実行した場合に比べ、より高速実行ができる。

【0049】また代理コンパイルサーバ10でJavaプログラムをコンパイルして要求元のクライアントに返送するため、クライアントが小規模のマシンでJITコンパイラをもつことができない場合でも、JITコンパイラをもつことのできるマシンと同様かそれ以上の実行スピードを得ることができる。

【0050】更に代理コンパイルサーバ10は、コンパイル済みのJavaプログラムをキャッシュ装置にキャッシングしておくことで、2回目以降の同じJavaプログラムの要求に対してはキャッシュ装置から読み出し

て返送することで直ちにクライアント側で実行でき、特にウェブサーバとクライアントとの間でプログラムやデータをやり取りするインタラクティブな動作環境において、コンパイルに要する時間を最小限にして、プログラムの立ち上がり時間やレスポンスを速くすることができる。

【0051】図8は、図7のステップS4における図3のコンパイル装置28のJavaプログラムのコンパイル処理の詳細である。図8のコンパイル処理にあつては、図5に示した制御メモリ26を利用し、要求のあったクライアント名から、その実行形式を認識する。

【0052】このときコンパイル装置28には、図6のようにクライアントの実行形式に適合した複数のJITコンパイラ38-1、38-2、38-3が準備されていることから、ステップS1で認識した実行形式に適合したJITコンパイラを選択してJavaプログラムのコンパイルを実行する。

【0053】図9は、図7のステップS4におけるJavaプログラムのコンパイルの他の実施形態のフローチャートである。このコンパイル処理にあつては、ステップS1でクライアントからのプログラムの要求メッセージを解析し、クライアントの実行形式を認識する。この場合には図5のような制御メモリ26は不要となる。次にステップS2に進み、ステップS1で認識したクライアントの実行形式に適合したJITコンパイラを選択してJavaプログラムのコンパイルを実行する。

【0054】図10は、図7のステップS4におけるコンパイル装置18のコンパイル処理の他の実施形態であり、この実施形態にあつては、代理コンパイルサーバ10に接続可能なクライアントマシンの全ての実行形式に適合したネイティブコードに翻訳してクライアント側に返送するようにしたことを特徴とする。

【0055】即ち、ステップS1で、要求を行ったクライアントとは直接関係せずに予め決められている特定のクライアントの実行形式を設定し、ステップS2で、この実行形式に適合したJITコンパイラを選択して、そのネイティブコードにコンパイルし、ステップS3で、コンパイルした実行モジュールをバッファに格納する。

【0056】続いてステップS4で全実行形式のコンパイル処理の終了の有無をチェックし、済んでいなければステップS1に戻り、次のクライアントの実行形式を設定し、同様に実行形式に適合したJITコンパイラを選択してネイティブコードにコンパイルし、ステップS3で実行モジュールをバッファに格納する。

【0057】ステップS4で全てのクライアントの実行形式のコンパイル処理が終了すると、そのときバッファに格納している各実行形式に適合したネイティブコードの実行モジュール、即ちコンパイル済みJavaプログラムを要求元のクライアントに返送する。

【0058】このようにして複数種類の実行形式のネイ

イル済みのコード長が長くなるネイティブコードのJavaプログラムについてはキャッシュ装置12に保存しないようにすることで、ハードディスクに使用するキャッシュ装置12としての容量を低減できる。

【0074】図14は、図13のコンパイル前のJavaプログラムのみをキャッシングする場合の図3の代理コンパイルサーバ10の処理動作のフローチャートである。この実施形態にあつては、ステップS1でクライアントからのプログラム要求を受け取ると、ステップS2で、要求されたJavaプログラムがキャッシュ装置12にキャッシングされているか否かチェックする。

【0075】キャッシングされていないならば、ステップS3でネットワーク18上のウェブサーバ20にプログラム要求を行ってJavaプログラムをロードし、ステップS4でキャッシュ装置12にJavaプログラムを保存するキャッシングを行った後に、ステップS5でコンパイル装置28によりJavaプログラムを要求元のクライアントの実行形式の適合したネイティブコードのJavaプログラムにコンパイルし、ステップS6で、コンパイル済みのJavaプログラムをクライアントに返送する。

【0076】このコンパイル済みJavaプログラムをキャッシングしない実施例にあつても、コンパイル前のJavaプログラムをキャッシングしておくことで、クライアント側の要求に対し、キャッシングしているJavaプログラムについてはネットワーク18上のウェブサーバ20に要求することなく直ちにコンパイル装置28でネイティブコードにコンパイルして要求元のクライアントに返送することができ、その都度、コンパイルを行う分、図7の処理に比べ時間はかかるが、キャッシュ容量を少なくできる分だけ、代理コンパイルサーバ10のハードウェア構成を簡略化できる。

【0077】図15は、図13のコンパイル前のJavaプログラムのみをキャッシングする場合について、図11と同様に、クライアントからの要求に対しプログラムをコンパイルして返送した際に、アプレット40に対するクライアントファイル42-1~42-4のダイナミックリンク45を解析して、次にクライアントが要求するJavaプログラムを予測してコンパイルする処理を適用したフローチャートである。

【0078】この図15の処理にあつても、ステップS1~S6は図14と同じであるが、クライアント要求プログラムをコンパイルして返送すると、ステップS7で図1のようなダイナミックリンク45の解析に基づき、次にクライアントが要求するプログラムを予測し、ステップS8で予測プログラムをキャッシングしているか否かチェックし、キャッシングしていないならば、ステップS9でネットワーク18上のウェブサーバ20から予測プログラムをロードし、ステップS10でキャッシュ装置12にキャッシングする。

【0079】ステップS8で予測プログラムがキャッシングされていれば、ステップS9、S10の処理はスキップする。この場合にも、クライアント側からあるJavaプログラムが要求されてコンパイル後に返送されると、次に要求されるJavaプログラムを予測して予めキャッシングしておくことで、実際に予測が当たった場合にはクライアント要求に対し代理コンパイルサーバ10より直ちに要求プログラムをコンパイルして応答することができ、ネットワーク上のウェブサーバ20のアクセス分だけJavaプログラムの実行時間を短縮することができる。

【0080】尚、上記の実施形態は、代理コンパイルサーバ10に設けたコンパイル装置28としてJITコンパイラを例にとるものであったが、Javaプログラムをクラス単位で翻訳する高速バイトコードコンパイラを設けてもよい。高速バイトコードコンパイラは、クライアント側がいつも同じマシンで動作するスタンドアロンアプリケーションの場合に適している。

【0081】またコンパイル装置10にメソッド単位での翻訳と最適化を行う機能修正、更新が容易なアプリケーションに適合したJITコンパイラと、機能が固定されたアプリケーションに適した高速バイトコードコンパイラの両方を設け、クライアント側のアプリケーション環境に応じて選択するようにしてもよい。

【0082】また上記の実施形態は、ネットワーク上の仮想マシンコンピュータプログラムとしてJavaプログラムを例にとるものであったが、ネットワーク上のプログラムをコンパイルしてクライアント側で実行する形態であれば適宜のシステムにそのまま適用できる。

【0083】更に本発明は、コンピュータネットワークに設けた代理コンパイルサーバ10そのものを対象としており、具体的には図16のようなプロキシサーバにJITコンパイラの機能を持たせることで実現できる。

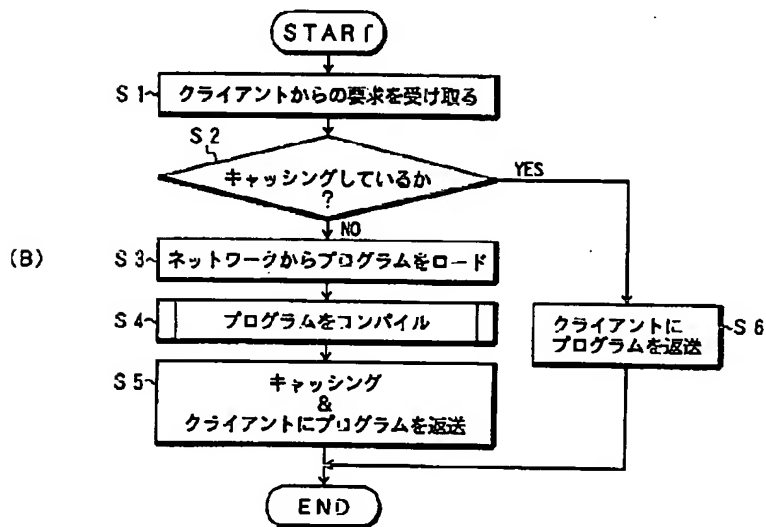
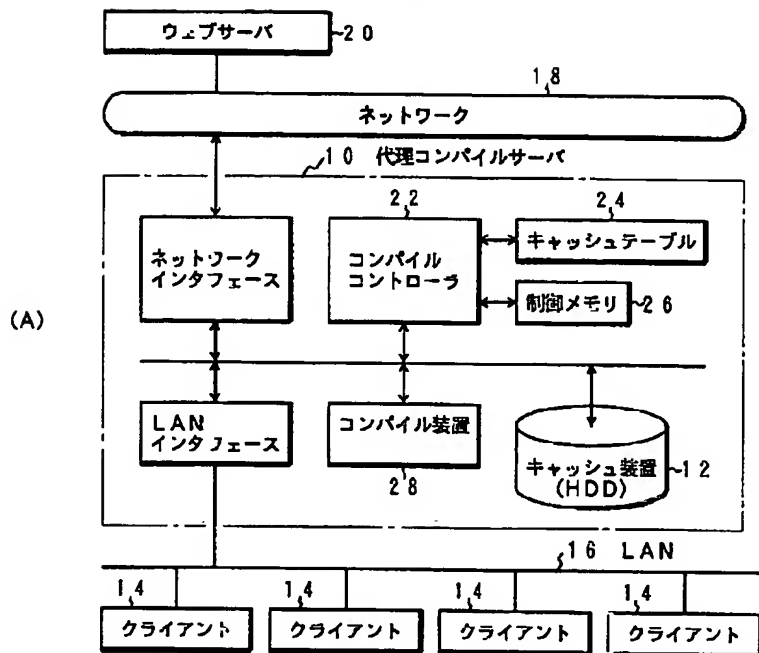
【0084】

【発明の効果】以上説明してきたように本発明によれば、ネットワークの代理サーバに仮想マシンコンピュータプログラムのコンパイル機能を付加して代理コンパイルサーバとしたことで、ネットワーク上の仮想マシンコンピュータプログラムを実行するクライアントの負担を減らし、ネットワークコンピュータシステムの下でクライアントがアプリケーションプログラムを高速実行することができる。

【0085】また代理コンパイルサーバにキャッシング機能を設け、クライアントの要求に対しキャッシングしているネイティブコードにコンパイル済みの仮想マシンコンピュータプログラムを返送することで、クライアントからの要求に対するコンパイル済みプログラムの返送が短縮でき、ネットワーク上のプログラム要求から実行までの時間が短縮され、結果としてクライアントにおけるアプリケーションプログラムの実行を更に高速化でき

【図1】

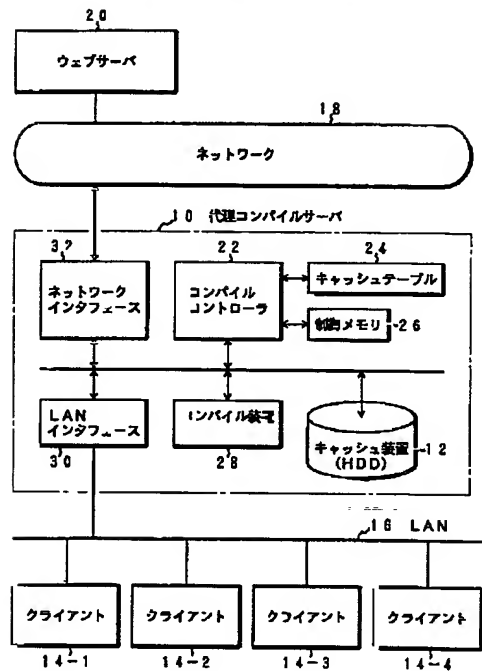
本発明の原理説明図





【図3】

本発明のシステム機能のブロック図



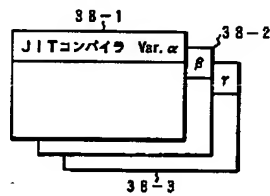
【図5】

図3の代理コンパイルサーバに設けた記憶メモリの説明図

26 記憶メモリ	
クライアント名	実行形式
A	$\alpha$
B	$\beta$
C	$\gamma$
D	$\alpha$

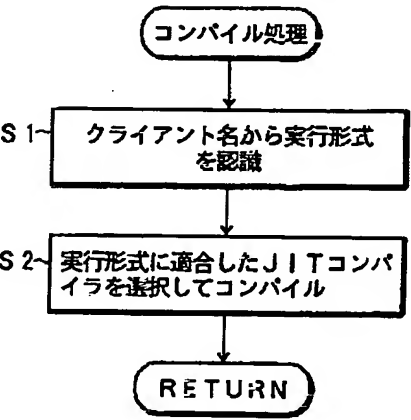
【図6】

図3の代理コンパイルサーバに設けたコンパイル装置の説明図



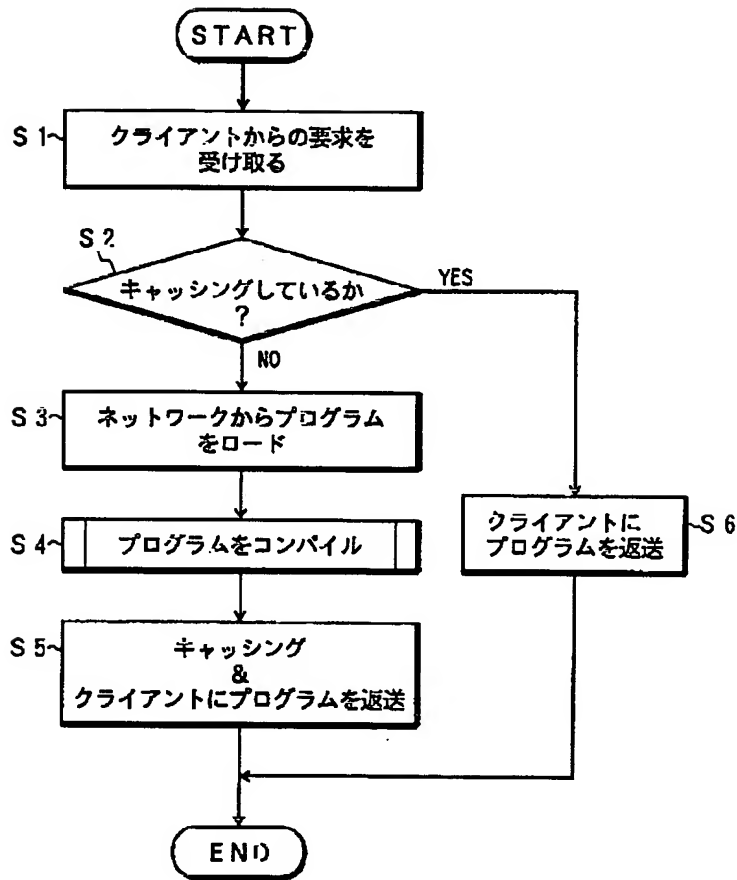
【図8】

クライアント名から実行形式を認識して行う  
図1のコンパイル処理のフローチャート



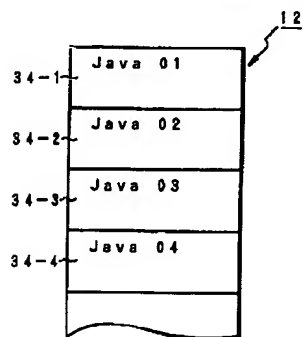
【図7】

図3の代理コンパイルサーバの処理動作のフローチャート



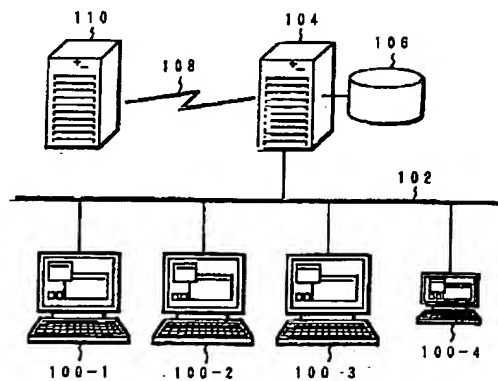
【図13】

ネットワークから受けたプログラムをそのまま保存する図3の代理コンパイルサーバに設けたキャッシュの説明図



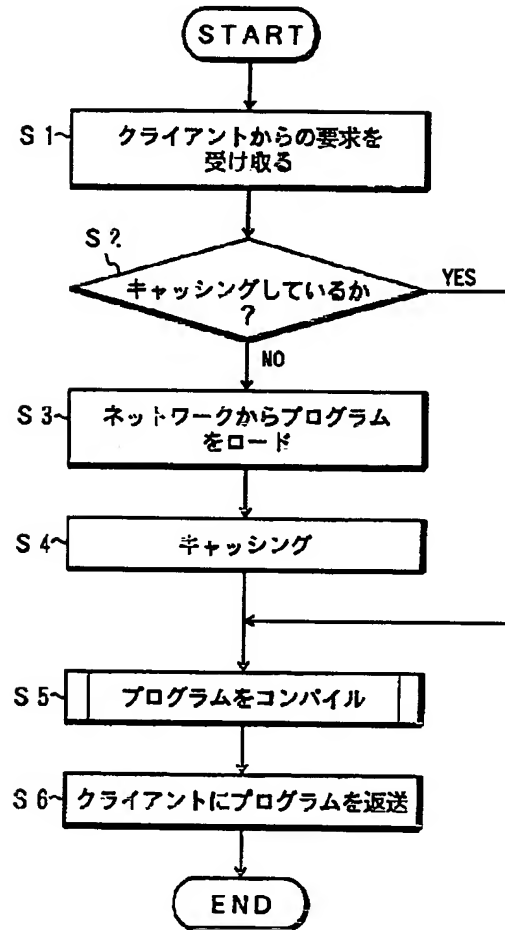
【図16】

従来システムの説明図



【図14】

図13のキャッシュを用いた図3の代理コンパイルサーバの  
処理動作のフローチャート



【図15】

クライアントの要求プログラムを予測して図13のキャッシュに保存する

図3の代理コンパイルサーバの処理動作のフローチャート

